Continuation Sheet: Issues to be Discussed.

Rejection of Claims in view of Wagner

Brief Description:

Wagner discusses formal methods and more particularly a technique that generates "constraints" or predicates upon analysis of the source code. Wagner observes source code statements and determines corresponding pre-defined constraints or predicates for the statements. The generated constraints are subjected to a "constraint solver" to determine if the source code has inherent pre-defined vulnerabilities. This in short is a system that checks the truth or falsity of the generated predicates or constraints (i.e., the constraints are proposition statements to be tested). There is mention of control flow and data flow (this was cited by the examiner) but the paper clearly considers control flow and data flow as other vehicles for generating yet more constraints to be tested. It does not mention control flow and data flow as a means of re-architecting Wagner's entire approach or deviating from the constraint-based approach.

The invention uses a different technique than Wagner. In short, the invention transforms the source code into a language independent intermediate representation that models the executable or operational semantics of the source code (e.g., including the control flow and data flow of the program). The invention subjects the intermediate representation/model to a virtual, or symbolic, execution to analyze the code as if it were executing. This is done to identify and determine vulnerabilities. It does not use constraints or predicate methods – unlike Wagner. We tried to capture this distinction in the prior amendment but this was subsequently determined to be unsatisfactory. We'd like to capture the distinction more clearly to everyone's satisfaction and to make sure the distinction is properly understood.